What Is Claimed Is:

1. A method for determining a load of a computing element on which a computer program is executed, the computer program being subdivided into a plurality of tasks, each of the plurality of tasks including at least one process, the method comprising:

(A) selecting a time interval such that at least one of the plurality of tasks is started and ended during the time interval;

(B) determining a running time of the at least one of the plurality of tasks during the time interval after a completion of one of the at least one of the plurality of tasks and each of the plurality of tasks; and

(C) if the at least one of the plurality of tasks was interrupted by another one of the plurality of tasks, subtracting a running time of the other of the plurality of tasks from a running time that includes the running time of the at least one of the plurality of tasks and the running time of the other of the plurality of tasks.

2. The method according to claim 1, wherein:

the time interval is selected such that at least two of the plurality of tasks are started and ended during the time interval, and

each running time is determined in a sequence in which each task is completed.

3. The method according to claim 2, wherein:

a variable Unterbr is set to zero before step (A),

the running time of each of the plurality of tasks when completed is determined using the equation $t_{runningtime} = t_{end} - t_{start} - (Unterbr_{end} - Unterbr_{start})$, and

a new value for the variable Unterbr is determined using the equation $Unterbr = Unterbr_{start} + (t_{end} - t_{start})$, t_{start} being a value of a time counter running during an execution of the computer program at a start of each of the plurality of tasks, t_{end} being a value of the time counter at an end of each of the plurality of tasks, $Unterbr_{start}$ being a value of the variable Unterbr at the start of each of the plurality of tasks, and $Unterbr_{end}$ being the value of the variable Unterbr at the end of each of the plurality of tasks.

4. The method according to claim 2, further comprising:

storing each running time of each of the plurality of tasks in a respective one of a plurality of individual memory locations of the computing element.

5. The method according to claim 4, wherein:

each of the plurality of individual memory locations includes a random access memory (RAM) memory location.

6. The method according to claim 2, further comprising:

determining the load of the computing element by forming an added value; and

setting the added value in proportion to the time interval after the time interval elapses, wherein the added value is formed by adding together the running times of each of the plurality of tasks.

7. A memory element, comprising:

a control program that when executed on a computing element causes the computing element to:

(A) select a time interval such that at least one of the plurality of tasks is started and ended during the time interval;

(B) determine a running time of the at least one of the plurality of tasks during the time interval after a completion of one of the at least one of the plurality of tasks and each of the plurality of tasks; and

(C) if the at least one of the plurality of tasks was interrupted by another one of the plurality of tasks, subtract a running time of the other of the plurality of tasks from a running time that includes the running time of the at least one of the plurality of tasks and the running time of the other of the plurality of tasks.

8. The memory element according to claim 7, wherein:

the memory element includes one of a read only memory, a random access memory, and a flash memory.

9. The memory element according to claim 7, wherein:

the computing element includes a microprocessor.

10. A computer program for execution on a computing element, wherein the computer program when executed causes the computing element to:

(A) select a time interval such that at least one of the plurality of tasks is started and ended during the time interval;

(B) determine a running time of the at least one of the plurality of tasks during the time interval after a completion of one of the at least one of the plurality of tasks and each of the plurality of tasks; and

(C) if the at least one of the plurality of tasks was interrupted by another one of the plurality of tasks, subtract a running time of the other of the plurality of tasks from a running time that includes the running time of the at least one of the plurality of tasks and the running time of the other of the plurality of tasks.

11. The computer program according to claim 10, wherein:

the computing element includes a microprocessor.

12. The computer program according to claim 10, wherein:

the computer program is stored on a memory element.

13. The computer program according to claim 12, wherein:

the memory element includes a flash memory.

14. A device for determining a load of a computing element on which a computer program is executed, the computer program being subdivided into a plurality of tasks, each of the plurality of tasks including at least one process, comprising:

(A) an arrangement for selecting a time interval such that at least one of the plurality of tasks is started and ended during the time interval;

(B) an arrangement for determining a running time of the at least one of the plurality of tasks during the time interval after a completion of one of the at least one of the plurality of tasks and each of the plurality of tasks; and

(C) an arrangement for, if the at least one of the plurality of tasks was interrupted by another one of the plurality of tasks, subtracting a running time of the other of the plurality of tasks from a running time that includes the running time of the at least one of the plurality of tasks and the running time of the other of the plurality of tasks.

15. The device according to claim 14, wherein:

the time interval is selected such that at least two of the plurality of tasks are started and ended during the time interval, and

each running time is determined in a sequence in which each task is completed.